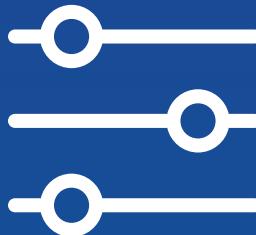


**50**

---

**TIPS**

**TO IMPROVE  
USER INTERFACE**



by Victor Ponamariov



# Contents

Foreword .....	8
----------------	---

## Typography

Put headlines closer to their respective area.....	9
Avoid justifying text in narrow columns.....	10
Break description text into a list of features .....	11
Make links look like links.....	12
Use proper line height.....	13
Center text container .....	14

## Forms

Don't hide form tips .....	15
Place input labels correctly.....	16
Use progress elements in complex forms.....	17
Show password rules right away.....	18

Require fewer fields.....	19
Use reasonable width of inputs.....	20
Use labels instead of relying on placeholders.....	21
Don't use dropdowns if the number of options is small .....	22
Replace default file inputs with custom ones.....	23
Autofocus the first input.....	24

## Validation

Autoscroll to the first error in large forms.....	25
Help users fill the form without errors .....	26
Consider positive as well as negative feedback.....	27
Show validation errors in the right place.....	28

## Focus and Contrast

Use only one primary button.....	29
Put an overlay on images for better text contrast .....	30
Check text contrast .....	31

## **Grouping**

Group form elements.....	32
Group logically related elements in the layout.....	33
Keep dangerous actions apart .....	34

## **Visuals**

Consider removing borders .....	35
Use proper colors for table borders/striping.....	36
Watch your shadows.....	37
Dropdown trick.....	38

## **Usability**

Put frequently used options on top.....	39
Add the copy to the clipboard button .....	40
Don't use complex forms in modals.....	41
Increase the clickable area .....	42
Don't rely on dots in gallery sliders.....	43

Consider removing confirmation modals .....	44
Label your icons.....	45
Provide further instructions.....	46
Indicate where the user is.....	47

## Loading States

Leverage empty states.....	48
Use skeletons .....	49
Button loading state .....	50
Don't show loader right away.....	51
Smart/Progressive loaders .....	52

## Misc

Icon inconsistency .....	53
Horizontal menu layout for links.....	54
Submitting verification code .....	55
Large data .....	56

Give users a visual hint that there is more content.....	57
Don't forget to put the post date .....	58

# Foreword

Hi there!

Recently I took a challenge. [I tweeted one UI tip per day for consecutive 50 days.](#)

This way, I was trying to improve my UI/UX skills. And provide some value for my followers.

Now I've put them all into a small book you're reading right now. It is my first try to write something. I'm pretty sure that there are some mistakes.

Remember. Most things in our world are opinionated. The book is not an exception.

These tips are recommendations, not strict rules. Some of them are less opinionated, some of them more. But eventually, the only advice you should follow is: be reasonable.

Victor.

## Put headlines closer to their respective area

The proximity law states: objects that are close to each other tend to be grouped together. The principle helps users follow and organize information faster and more easily.



In the first picture the headline has the same distance between the first and the second paragraphs.

In the second picture, the headline is placed closer to the text to which it is related.

If you keep your headlines closer to their related text blocks it will reduce the cognitive load.

Avoid justifying text in narrow columns

Justifying text in narrow columns is considered a bad practice.  
The reason is justified text creates the "rivers" effect.

Rivers are the white gaps in the text when there is too much space between words on consecutive text lines.

It makes the text look unbalanced and harder to read.

*Donec aliquet mi orci, a varius metus rhoncus eu.*

Donec ultricies lobortis libero.  
Nullam vitae rhoncus nisi. Nullam at  
euismod tortor, a suscipit nulla.  
Mauris id est et mauris rutrum  
tincidunt.

Instead align text to the left.

Donec ultricies lobortis libero.  
Nullam vitae rhoncus nisi. Nullam at  
euismod tortor, a suscipit nulla.  
Mauris id est et mauris rutrum  
tincidunt.

## Break description text into a list of features

Almost nobody will read your site entirely. People can barely focus their attention for a few minutes. Instead, they **scan** sites.

What you can do is **hook user attention**.

One of the nice tricks is to use lists. You should list what is listable.

This way you can also reduce the amount of text by removing commas, conjunctions etc.

---

Arrange your photoshoots, client meetings, private events and automate appointment reminders to reduce no-shows.

Do everything on the go and recuve manual admin work, so you can focus on your photography business

---

Arrange your photoshoots, client meetings and private events

Automate appointment reminders

Reduce manual admin work

Focus on your photography business

## Make links look like links

Here is a screenshot from Twitter analytics. Can you guess if there is a link to the tweet?



The link is the date of the tweet, which is impossible to guess.

First, when you click on a date you probably expect to get all tweets you had that day.

Second, the link doesn't look like a link. It's gray and small.

Make sure users understand what are links without hovering them.

  
Lorem ipsum **dolor** sit amet,  
consectetur adipiscing elit. Fusce  
hendrerit **viverra** pellentesque.

  
Lorem ipsum **dolor** sit amet,  
consectetur adipiscing elit. Fusce  
hendrerit **viverra** pellentesque.

## Use proper line height

Text line height should be proportional. The bigger the font size is, the smaller the line height should be.

For headlines, the multiplier is ~1 - 1.4. For other text ~1.3 - 1.6

	Line height
<b>Sed quam urna, vulputate sit amet enim sit amet</b>	100%
Mauris et quam vitae ligula facilisis rutrum vel ut elit. Proin tempor erat et urna congue posuere.	100%
Pellentesque metus augue, congue eu quam pretium, sagittis hendrerit nibh. Proin quis scelerisque ligula. Sed vel turpis cursus, posuere eros bibendum, dictum mauris. Aliquam in blandit tellus, sit amet vulputate ante.	100%
	Line height
<b>Sed quam urna, vulputate sit amet enim sit amet</b>	110%
Mauris et quam vitae ligula facilisis rutrum vel ut elit. Proin tempor erat et urna congue posuere.	130%
Pellentesque metus augue, congue eu quam pretium, sagittis hendrerit nibh. Proin quis scelerisque ligula. Sed vel turpis cursus, posuere eros bibendum, dictum mauris. Aliquam in blandit tellus, sit amet vulputate ante.	150%

## Center text container

Do not stretch the text container to the full width. Otherwise, your eyes will move in Z-pattern. It will be hard to read.



**Etiam** euismod et tellus sit amet pellentesque. Mauris in turpis et enim bibendum eleifend nec et metus. Quisque sagittis turpis at finibus consectetur.

## THIS IS HARD TO FOLLOW



*Etiam* euismod etiam euismod. Etiam euismod etiam euismod. Etiam euismod etiam euismod. Etiam euismod etiam euismod. Etiam euismod etiam euismod.

## Don't hide form tips

It looks like a good solution to hide hints under an icon because we can save some space. The UI will look cleaner.

But. The spacing is usually not a problem.

While the user will need to take many additional actions. He should hover every tooltip to find out what you wanted to say.

A better solution would be to display hints right away. For example, below inputs.

Red bar	Green bar
?	
Required	
?	
Leave empty if you don't have any	
	We will send SMS to this number

## Place input labels correctly

If you have a two-column layout, it would be best to place input labels near inputs. Otherwise, users' eyes will move in Z-patter, which is not recommended.

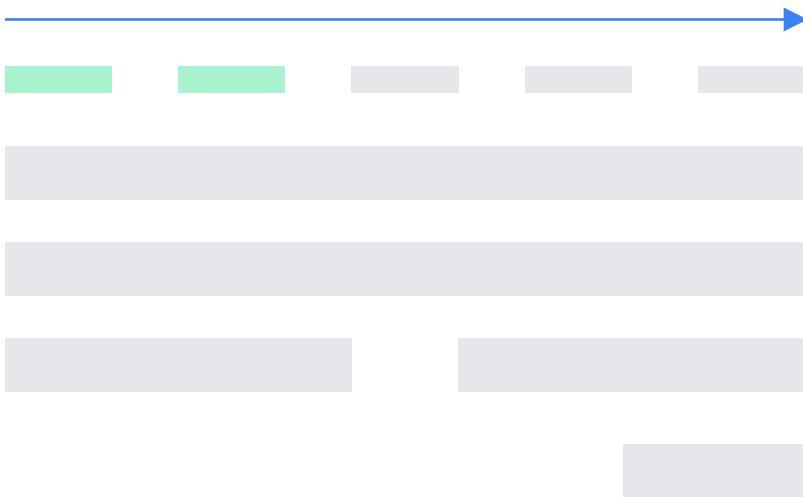
A diagram illustrating a two-column layout for form fields. On the left, labels are placed vertically: "First Name:", "Last Name:", "Email:", and "Password:". To the right of each label is a light gray rectangular input field. Three thin gray arrows point from the top of each label towards its corresponding input field. A thicker gray arrow points from the bottom of the "Password:" label towards its input field, indicating the direction of user eye movement.

A diagram illustrating a two-column layout for form fields. On the left, labels are placed vertically: "First Name", "Last Name", "Email", and "Password". To the right of each label is a light gray rectangular input field. A single thick gray arrow points downwards from the bottom of the "Password" label towards its input field, indicating the direction of user eye movement.

## Use progress elements in complex forms

If you have a large complex form, break it into sections and display them one at a time. At the same time, place a progress element that highlights the current step.

That will reduce the reluctance users might have. Also, there is [the goal gradient effect](#) that may motivate them to finish completing the form.



## Show password rules right away

If your password conforms to quite a few rules, show what you require from the user right away.

Please do not wait until the user hits the submit button to show him a set of errors.

Also, showing the requirements is not enough. When the user fills the password form, highlight the satisfied rules.

A screenshot of a password input field. The input field contains six dots ('.....'). To the right of the input field is a green horizontal bar. Below the input field, there is a 'Password' label, a 'Show' button with an eye icon, and a list of five validation rules. The first four rules have green circular markers next to them, while the fifth rule has a grey circular marker.

Rule	Status
One lowercase character	Green
One uppercase character	Green
One number	Green
8 characters minimum	Grey

*This is how Mailchimp does it*

## Require fewer fields

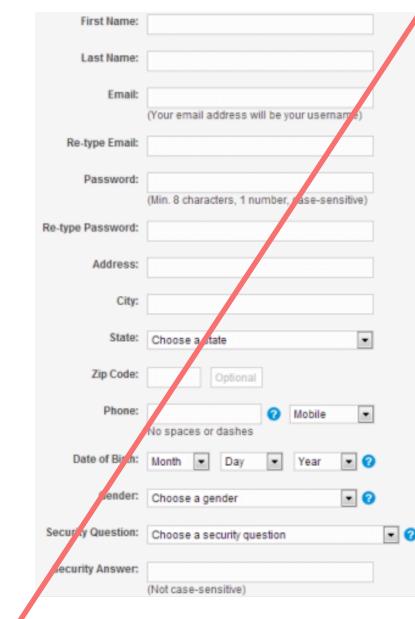
The more fields you require, the more significant becomes the barrier for users to fill the form.

You don't need the user name to let him use your app.

You don't need the password confirmation since he can restore it via reset password form.

You probably don't even need to verify user's email to let them try your app out.

You can allow them to use the features that don't require email verification. And only ask it when they want to do something crucial that requires verified email.



A screenshot of a complex user registration form. The fields include:

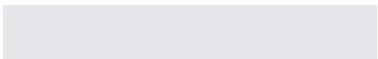
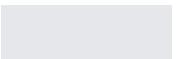
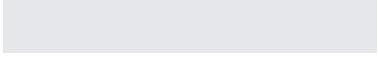
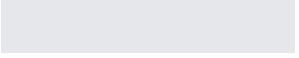
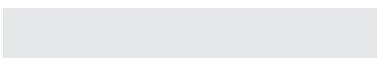
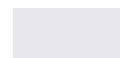
- First Name: [Input Field]
- Last Name: [Input Field]
- Email: [Input Field]  
(Your email address will be your username)
- Re-type Email: [Input Field]
- Password: [Input Field]  
(Min. 8 characters, 1 number, case-sensitive)
- Re-type Password: [Input Field]
- Address: [Input Field]
- City: [Input Field]
- State: [Select Box] Choose a state
- Zip Code: [Input Field] Optional
- Phone: [Input Field] Mobile  
No spaces or dashes
- Date of Birth: [Select Box] Month Day Year  
Month Day Year
- Gender: [Select Box] Choose a gender
- Security Question: [Select Box] Choose a security question
- Security Answer: [Input Field]  
(Not case-sensitive)

## Use reasonable width of inputs

Do not try to fill the remaining space on your screen. Your forms don't have to be 100% wide.

The size of inputs can determine what type of content you expect from the user.

For example, a zip code or phone number cannot have a hundred characters.

	
Zip Code	Zip Code
	
Phone Number	Phone Number
	
Date	Day      Month      Year
	  

## Use labels instead of relying on placeholders

Don't use placeholders as labels.

While the user is filling the form, he might forget what you asked him. To see the placeholder text, the user has to erase the content in the input.

Plus, having labels is better for accessibility reasons.

In rare cases, though, you can omit labels. In a sign-in form, you have only two inputs, so it's okay to use placeholders.

Placeholder	Label
First Name	First Name
Last Name	Last Name
Email	Email
Password	Password

## Don't use dropdowns if the number of options is small

If you have a dropdown with up to 5-7 options, then display all the options right away. Do not hide them under the dropdown.

Users will need to make more clicks to select the option they need. You have plenty of options.

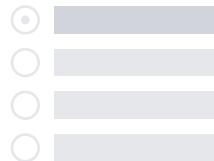
1. Radio buttons
2. Clickable tags/buttons
3. Clickable cards
4. Checkboxes (for multiple selections)



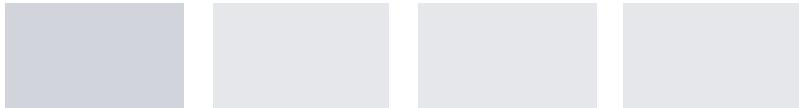
Clickable tags/buttons



Radio buttons



Clickable cards (may contain radio buttons)

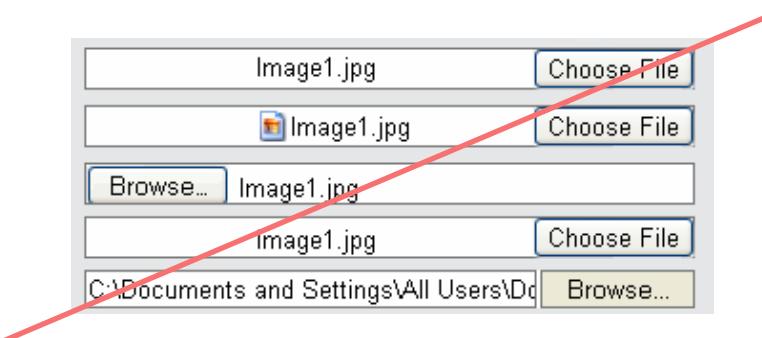


## Replace default file inputs with custom ones

Default file inputs look different in different browsers. Plus, they have poor usability.

They don't have many things like image preview, size validation, asynchronous uploading, etc.

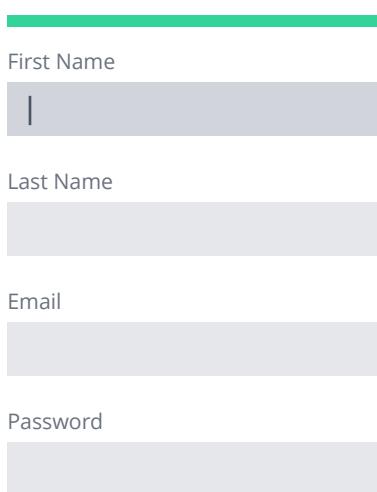
Instead of using inconsistent default browser inputs, you should use a custom one.



## Autofocus the first input input

When building a user-friendly interface, we aim to reduce the number of actions users need to take. Ideally, we shouldn't have any interface at all.

A simple but nice trick is to autofocus the first input in the form. This way user won't need to do it manually.



First Name

|

Last Name

Email

Password

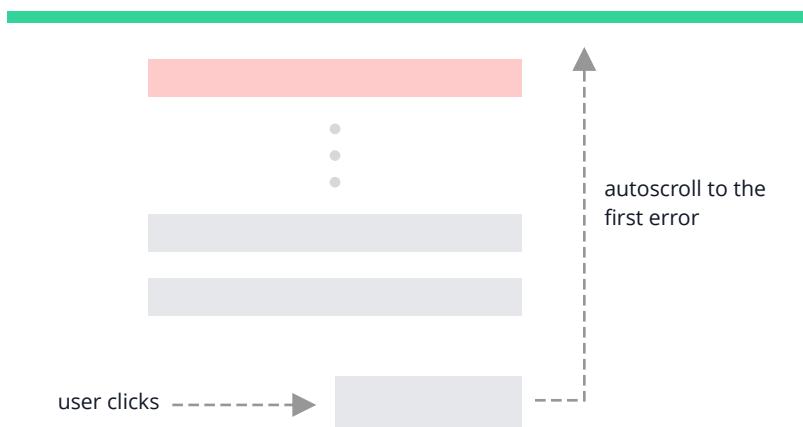
## Autoscroll to the first error in large forms

Sometimes the size of a form can exceed the monitor viewport.

When the user scrolls down to the submit button and presses it, some of the inputs placed above can be filled incorrectly. But the user can't see them.

On the one hand, you have correctly working validation. On the other hand, when the user clicks the submit button, nothing happens for him.

A possible solution is to move users to the first invalid input that contains an error description.



## Help users fill the form without errors

Even though you should have form validation, it'd be the best case if your users fill the form correctly on the first try. There are some tricks on how you can do it.

1. Reduce the number of required fields
2. Pre-fill data if you can do so. For example, you can try to guess the user's country, language, location, device, etc.
3. In login/signup forms provide social authentication methods.
4. Use masks for such fields as dates and phone numbers so that users don't guess the correct format.
5. Visually show how long the content should be. For example, don't make the input very wide for zip codes since it's not realistic.
6. Limit the symbols that the input accepts. For zip code it's reasonable to allow inputting only numbers.

## Consider positive as well as negative feedback

If you have a complex form, it might be a good idea to show users not only when he fails but also when he fills an input correctly.

It gives users some assurance.

It might not be necessary for small forms like login. But if the form consists of dozens of inputs with some tricky data required, show a green tick if the field is filled correctly.

Email

john@gmail.com



Card Number

1111 2222 3333 4444



Phone Number

+1-111-30-4



Confirm

## Show validation errors in the right place

One of the best choices is to place validation errors near inputs. If you show an error that way, the user will be in the context of what he's filling in.

While if you show it somewhere else, the user will have to match the text and the input.

Plus, if you group errors in one place (e.g., at the bottom), it might end up with many lines that's hard to read.



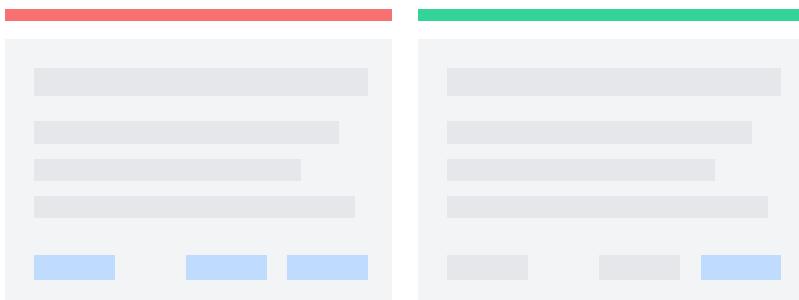
## Use only one primary button

You should show users one logically completed piece of information at a time.

If you have a form, it should be related to one function and has one primary button. The primary button usually has the brand color.

One of the common mistakes is having too many primary buttons.

Instead, you should have one primary button, while other buttons/links should be secondary.



## Put an overlay on images for better text contrast



Sometimes you need some text on an image. But you may not know what the picture will be.

Just putting text won't work. You have a few solutions here.



First, you can put a black overlay on the image with some opacity.



Another solution is to add a background to the text, so that the contrast looks good.

## Check text contrast

There are three WCAG contrast levels: A, AA, and AAA. You're good if you have AA or AAA.

Having low contrast is a widespread mistake. A vast number of sites fail to conform even to the AA level.

You can quickly check the contrast level with chrome developer tools.

AA failed (2.3:1)

Lorem ipsum dolor sit amet,  
consectetur adipiscing elit.  
Morbi faucibus, dolor non  
dapibus elementum, magna  
tellus sagittis sem, sit amet  
dapibus odio nibh eget ligula.  
Fusce at quam elit.

AAA passed (9.36:1)

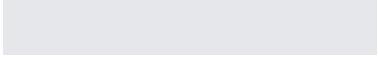
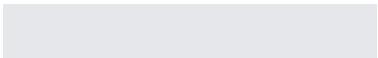
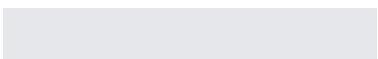
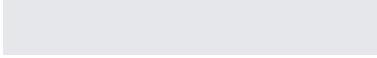
Lorem ipsum dolor sit amet,  
consectetur adipiscing elit.  
Morbi faucibus, dolor non  
dapibus elementum, magna  
tellus sagittis sem, sit amet  
dapibus odio nibh eget ligula.  
Fusce at quam elit.

## Group form elements

Proximity law states: "Objects that are near, or proximate to each other, tend to be grouped together."

A common mistake is when input labels have the same spacing between inputs. Without additional effort, you cannot easily say to which input the label belongs.

"Whitespace is like air: it is necessary for design to breathe." – Wojciech Zieliński

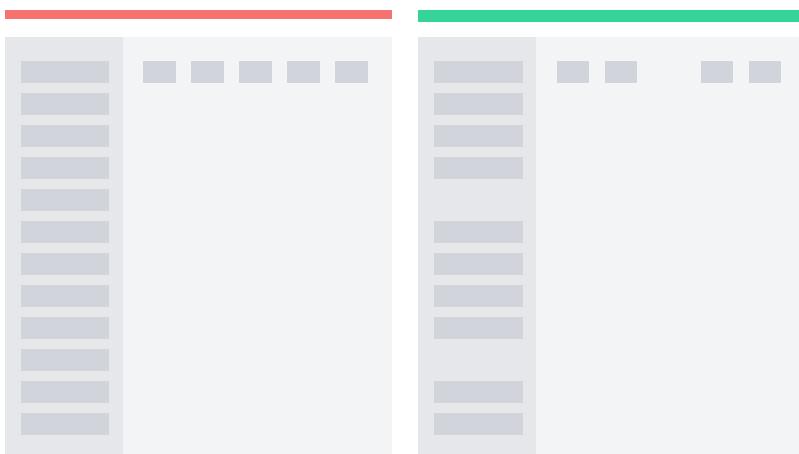
 First Name  Last Name  Email  Password 	 First Name  Last Name  Email  Password 
---	---

## Group logically related elements in the layout

Miller's law states that the number of objects an average person can hold in working memory is about seven plus or minus two.

You shouldn't take this literally, trying to fit elements in the magic number. But you should place logically related elements into groups and provide enough whitespace between them.

If your navigation has 10-15 links that are not divided into any logical groups, then it'll be more challenging for users to navigate through them.



## Keep dangerous actions apart

Actions that can lead to irreversible effects should be kept apart from frequently used ones.

Don't put such things as delete buttons, logout links, send message buttons near other actions.



*The power off button located directly above the delete key*

## Consider removing borders

Often user interface looks much cleaner without borders. When founders or developers work alone without a designer, they tend to overuse borders.

An alternative would be using background colors (without borders) or shadows. Or even don't use anything at all.



**Use proper colors for table borders/striping**

Two rules can help you make your table look better.

If you **don't** have many rows - omit bordering or striping them. Instead, enlarge the whitespace between rows.

If you have a massive table with many rows and columns, it will be hard to navigate through it. In this case, you can stripe your stripe or border it.

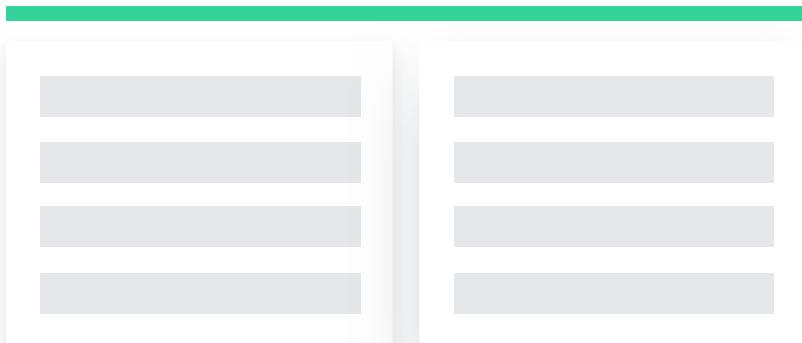
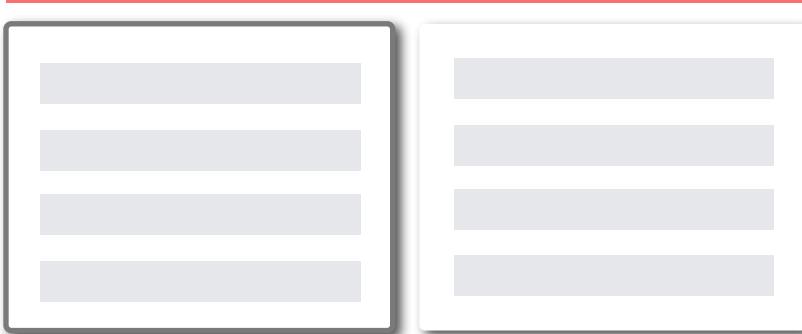
But pay special attention to the colors. They should be subtle and barely noticeable.

Otherwise, the table will look bad.

## Watch your shadows

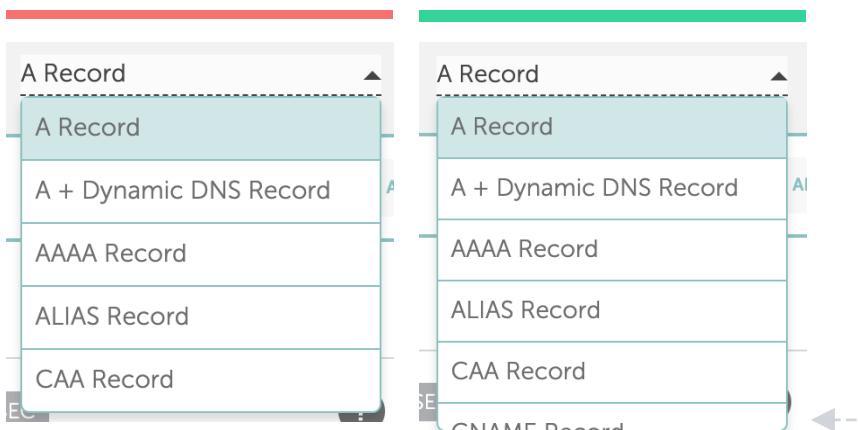
Your shadows should be soft and have a slight presence.

Also, it would be better to avoid using borders simultaneously since it might be hard to make them look good together.



## Dropdown trick

This one is easier to illustrate than describe. In the Namecheap DNS records editor, there is a dropdown for adding a new record.



In the first case, they have a dropdown with some options that exactly fit the dropdown.

But it's impossible to guess that there are, in fact, more options. Especially on MacOS, when the scrollbars are hidden.

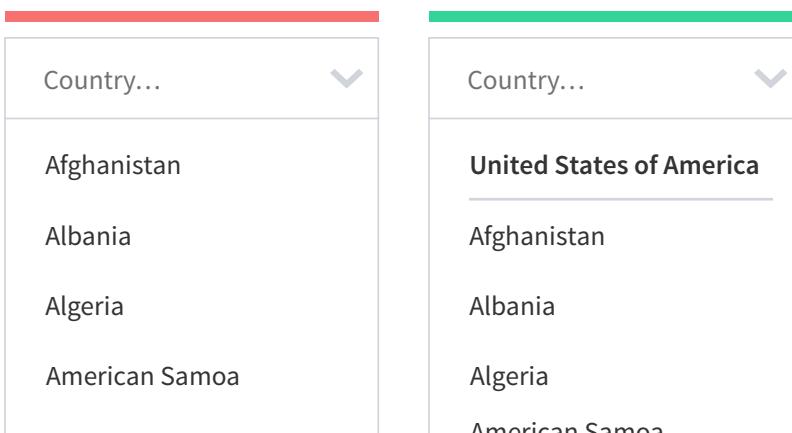
All you need to do in this case is increase the dropdown height to give users a clue that more options exist.

## Put frequently used options on top

The fewer actions the user should take to achieve his goal - the better. One of the tricks is to provide frequently used options right away.

If somebody needs another option, he will have to do more work. But by providing frequently used options, we can reduce the amount of effort for others.

You can also remember what users often select and then show it on the top. It perfectly suits search results.



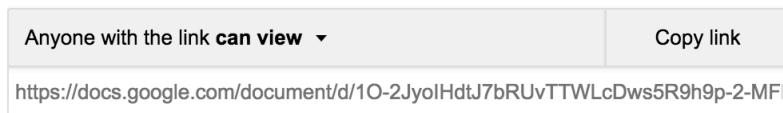
## Add the copy to the clipboard button

You should add the "copy to clipboard" button for links, images, tokens etc. In other words, for everything that users have to copy & paste manually.

Share with others

Get shareable link 

Link sharing on [Learn more](#)

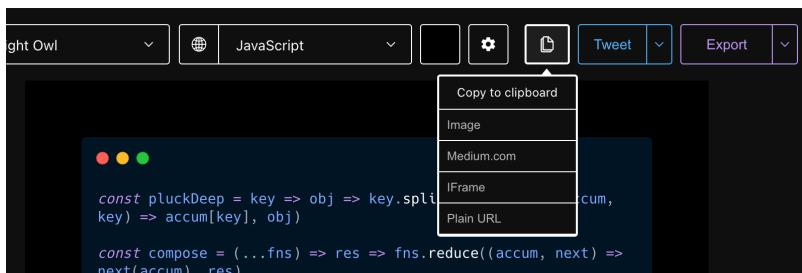


Anyone with the link **can view** ▾

**Copy link**

<https://docs.google.com/document/d/1O-2JyoIHdtJ7bRUvTTWLcDws5R9h9p-2-MF>

*Google drive document sharing*



*carbon.now.sh image copying to clipboard*

## Don't use complex forms in modals

Modals themselves [have a lot of disadvantages](#). But here is a specific case.

Imagine you have a table of entities and the "add new" button.

When you click the button, you have two options: redirect the user to a new page or show a modal.

In the latter, you might encounter the following problems:

1. It's harder to implement a specific URL that will show the form. Without the URL you cannot share it.
2. If the form doesn't fit the modal you'll have a scroll. It might lead to such things as a not-visible submit button.
3. The user can accidentally close the modal in the middle of filling it.
4. On mobile devices, modals take a bit more space than if we place the form on a separate page.

To sum up: avoid putting complex forms in modals and don't overuse them too much.

## Increase the clickable area

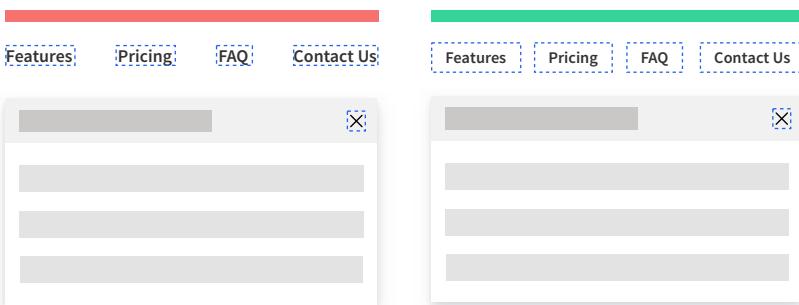
Fitt's law reads: The time to acquire a target is a function of the distance to and size of the target.

In other words, the bigger the object is, the easier it is to hit it.

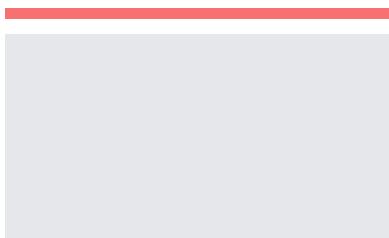
For small controls, you can add an invisible clickable area that will help users hitting them.

It is crucial on mobile devices.

The invisible area is easy to achieve by simply using paddings for HTML elements.



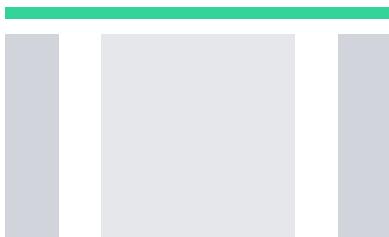
## Don't rely on dots in gallery sliders



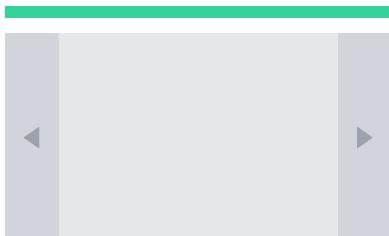
Dots have poor usability because it's hard to hit them.

They are completely unusable on mobile devices.

You have many other options.



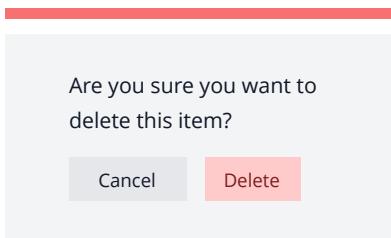
Display clickable part of previous / next images



Make left & right areas of the image clickable.

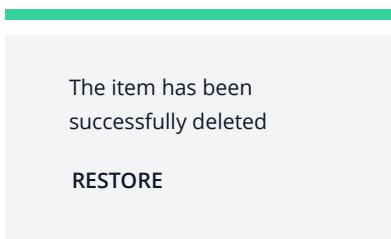
Pay attention that the clickable area should be big enough.

## Consider removing confirmation modals



Confirmation modals don't work. You almost automatically click yes, because it became our habit.

It's a problem if you cannot restore important information.



It would be much better to allow the user to restore deleted information.

In OS, when you delete a file, it goes to the trash bin at first.

But if the user asked something that cannot be undone, it's not an option.

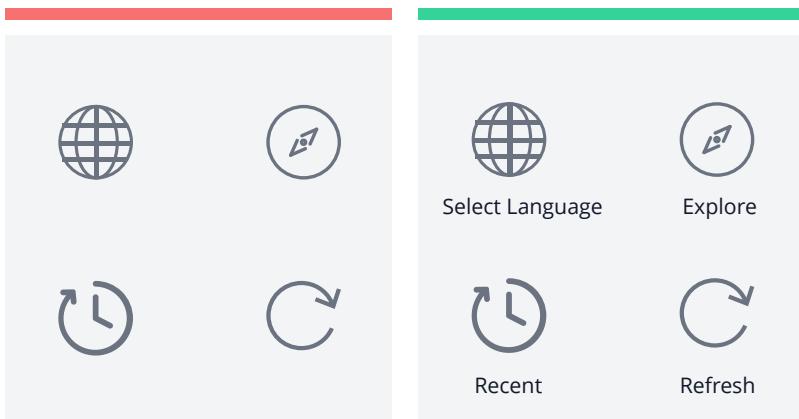
In this case, you should enhance your confirmation. You can ask your users to do some manual work. For example, by asking them to type something manually.

## Label your icons

Since there is no standard of what each icon means, you should label your icons.

The same icon can mean different things. Every developer and designer has his own opinion of what it should represent.

I found an [excellent article](#) explaining in detail why you should label your icons, so I'd better leave it here for further reading.



## Provide further instructions

You shouldn't leave users without any clue of what they should do next.

Some examples of doing it wrong:

An error occurs, and you show users what happened, but you don't tell them what to do next.

The user googled your site and clicked on the link, but instead, he got a 404 error. Page is not found. But you didn't give him instructions on what to do next.

User logged in to your complex app. And there is no onboarding or any hints of what he should do first.

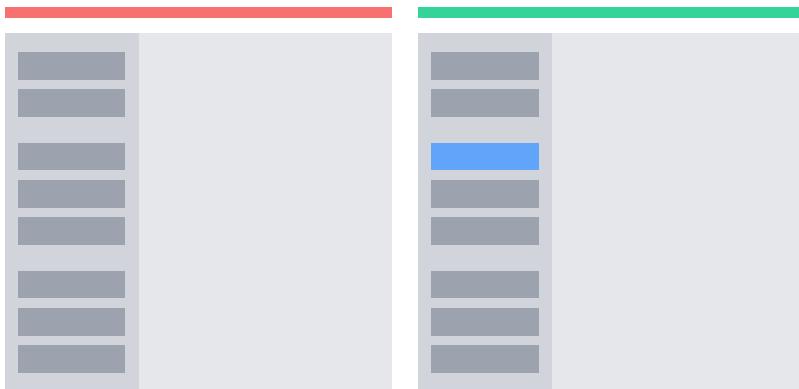
So the general rule is that users shouldn't be stuck without any clues of what to do next. There is a related heuristic called [user control and freedom](#).

## Indicate where the user is

Users should always know where they are at every moment they use your app.

Highlight navigation links. Add breadcrumbs if your site map is complex.

There is a related usability heuristic, “[Visibility of System Status](#)”



## Leverage empty states

When you need to display a list of records, a common pattern looks like this.

### Credit Cards

Add New

You don't have any cards

But if there are no records, you can use available space for a more informative and engaging message.



### No Credit Cards

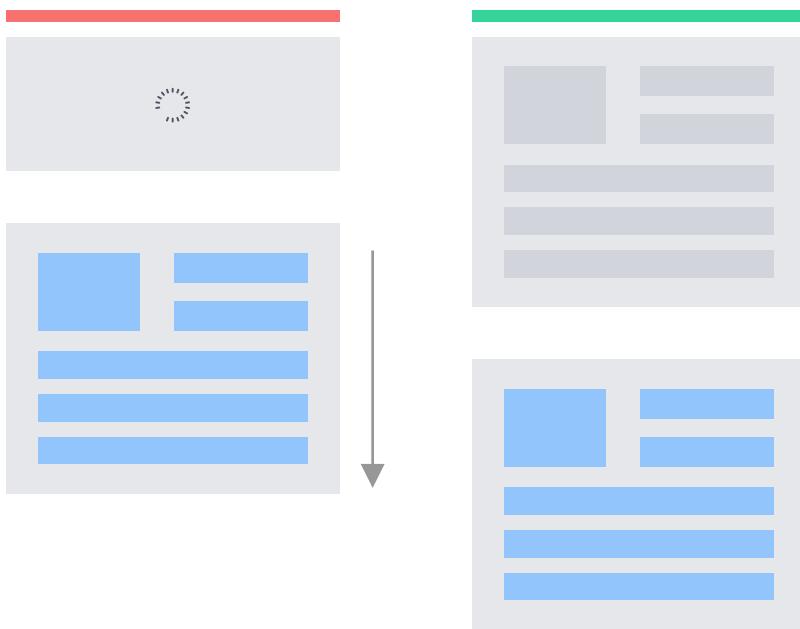
Please add some credit card to your account

Add a credit card

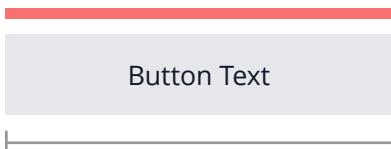
## Use skeletons

If you use loaders when loading big content, you'll have a layout shift after the content was loaded. While if you use skeletons, the height will stay the same.

It's best to use loaders in such things as buttons. At the same time, skeletons work well for layout / articles / large content.



## Button loading state



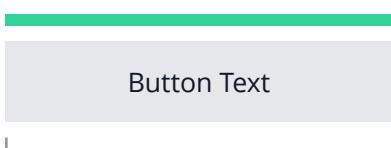
Buttons should keep the same width when they turn into the state of loading.



The layout shift happens when you hide button content and show loader instead.



The best solution here is to make the text transparent.



It will ensure that the button will have the same width because the content has not changed. It became transparent instead.



It will work even if you have localization and the user has changed the language.

## Don't show loader right away

Developers usually display the loader as soon as the content started loading. But in reality, the content may be loaded in traction of seconds, e.g., in ~0.2 seconds.

In this case, showing loaders would look like a flashing piece of content. Things will look even worse in the case of displaying a large piece of content, e.g., an article.

It would be best to wait sometime (e.g., 0.5 - 1 seconds) before showing loaders to avoid the layout shift.

## Smart/Progressive loaders

Sometimes it happens that the content is loading for a long time. It might be because of a slow connection or if you have heavy calculations on the backend side.

In this case, when the loader is animated, and nothing happens for 5-10 seconds, the user might not be sure if it's working or not.

Instead of showing the same loading animation, you can provide additional engaging content. For example, by saying, "Hmm, it takes more than usual time...", "We're almost here..." etc.

It will give users some assurance that the app is working and not broken.

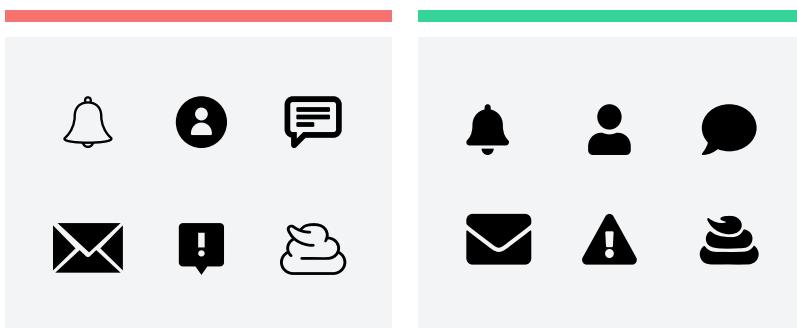
## Icon inconsistency

When you find some free icons on the web that belongs to different packs, it might turn out that they have different styling.

- Different stroke width
- Filled / Outlined
- Coloured / Non-coloured
- Minimalistic / Complex

It'd be better, of course, to use a pack that has some consistent styling.

But if you download a single icon that doesn't exist in the pack, make sure it conforms to the styling you already have.

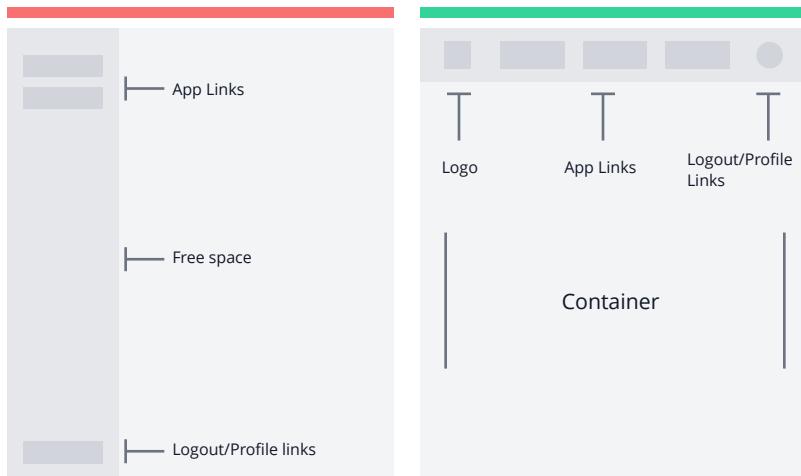


## Horizontal menu layout for links

If your app is simple or doesn't have yet many pages, it might be a good idea to use horizontal navigation instead of a vertical one.

The reason is that the vertical navigation takes all viewport height while horizontal one can be centered.

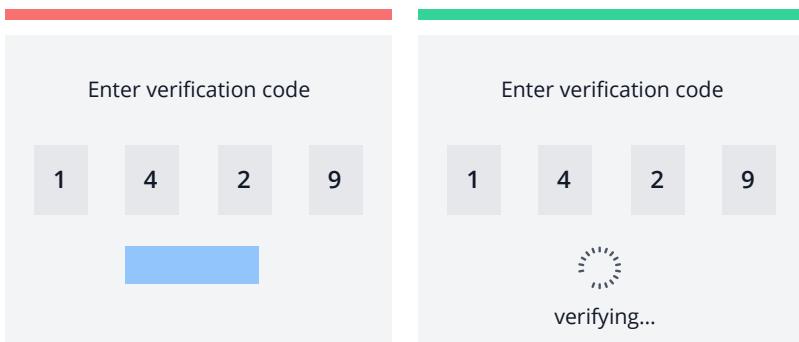
Also, horizontal navigation quite often has additional elements as logo and user menu.



## Submitting verification code

If your code is simple (e.g., has four digits), you might avoid asking the user to click the submit button.

The probability that the user will enter it incorrectly is small (especially if he copy-paste it from an email message). This way, you can reduce the number of required actions.



## Large data

A widespread case is when the design is made assuming that the content data won't be extensive: emails, user names, article titles - all that believed to be small.

But in reality, it happens very often that the content is large. User name suddenly consists of few words, article's title has few sentences.

In this case you the content should be handled appropriately. It might be:

- Hiding overflowing content and putting dots ellipsis
- Forbidding users to enter lengthy content by limiting the number of characters they can input
- Hiding part of the content and showing the user the "show more" button

The main thing is not to forget that the data can be lengthy.

## Give users a visual hint that there is more content

It is a pretty common trick to visually hint to users that there is more content than they see at this moment.



## **Don't forget to put the post date**

Our knowledge constantly evolves. The content written today might be irrelevant tomorrow.

When I google something related to programming and find an article without a date, there is no point in reading it because you cannot be sure that it is not outdated.

The article might've been written five years ago. It's a huge time range for programming.

The same applied to many areas of our life.

So, don't forget to put the post date.

## Thanks for reading!

If you have any questions, I'm [always available on Twitter](#).

Also, I share some UI/UX stuff there daily, so don't forget to check it out!